Software Integration Testing

Component: Software Integration Testing

Component Description:

The Software Integration Test (SIT) should be performed by a professional test group within the design activity, independent of the development group. The test group normally is comprised of experienced technical and functional software test analysts who work directly with application subject matter experts to develop test cases and test data that will be used throughout the testing process.

The primary scope of the Software Integration Test: validate that the software meets the user defined requirements, the technical system requirements, and application specific interface requirements. The method selected by the developers to integrate and turnover the software components to the testing group, will determine the approach to the test. It may be necessary to test only portions of an application until such a time that all the components have been integrated into a complete application and a full test of the application can be conducted.

The test cases (detail) for the Software Integration Test are designed based upon the requirement specification and incorporate both positive and negative testing. Apart from individual unit level testing and module testing, the Software Integration Test is a more comprehensive test of the software components than any test that follows. At the completion of this test component, test cases, test scripts, and test data should be available in support of the creation of functional test scenarios for the functional validation test that will be performed by customer representatives.

The Software Integration Test is conducted in a test environment which supports the creation and maintenance of test databases. These can be shared by multiple testers in a controlled manner to execute test cases which involve interaction of multiple software components or business functions. Typically, control of the test environment is directed by the test group.

SIT testing includes the following types of test:

1) Performance assessment
2) System interface testing
3) Functionality / user requirements testing
4) Operability testing
5) Security and control testing
6) Standards verification
7) Data conversion verification, and
8) Compliance requirements validation, as possible

Inputs:

1) Application Business Process Model (RD.010)
    a) Event Catalogue
        1. Name
        2. Description
        3. Frequency and condition under which it occurs
    b) Process Descriptions
        1. Description of process that is executed in response to each event
        2. Inputs
        3. Outputs
    c) Process step catalogue
        1. List of the process steps
        2. Person responsible for performing steps
        3. Degree of automation (manual, partial automation, full automation)
    d) Process Flow Diagram
    e) Process re-engineering list

2) Application Business Function Model (RD.030)
   a) Function Hierarchy Diagram
   b) Function Data Usage
      1. CRUD matrix
      2. Data flow diagram
3) Application Business Data Model (RD.040)
   a) Entities – the objects or things of significance that an organization wishes to hold information about
   b) Synonyms – the aliases or other names of the entities
   c) Attributes and domains – the key or descriptive properties of the entities
   d) Relationships – the way entities are related or associated to one another
   e) Unique identifiers – the combination of attributes and relationships that uniquely identify each instance of the entities
4) DCII specification
   a) Appendix D – DCII specification requirements
5) Test artifacts from unit/module testing
   a) Module Test Plan (TE.030)
   b) Unit test scripts / checklists and data (TE.040)
   c) Stubs / drivers
   d) Test results
      1. Conversion Module test results (CV.060)
      2. Unit / Module test results
6) Application Test and Evaluation Master Plan (TEMP)
7) Module Functional documentation (MD.040)
   a) Functional logic of the module
   b) Layouts
   c) Constraints
   d) Deviations of the standards for user interface
8) Module Technical documentation
   a) Module logic
   b) Detailed table and column usage
   c) Module parameters (in-going and out-going)
   d) Supporting modules
9) Application Interface specification (TA.010)
   a) Source system of data / target system for data
   b) Format
   c) Frequency
   d) Available date of test data
   e) Available date of production data
   f) Point of contact for interface data
   g) Name of interface files
   h) IP location data is transmitted to or extracted from
   i) Directory location data is transmitted to or extracted from
   j) Processing or messaging that occurs to insure accurate transmission
   k) Security constraints involved in the transmission
10) Detailed System Operational requirements (TA.050)
   a) Start-up and shut-down
   b) Security enforcement and handling and reporting of detected breaches or breach attempts
   c) Audit
   d) Error notification and reporting
   e) Backup
   f) Error recovery
   g) Fallback after disasters
   h) Archive and purge of old data
11) Security and control strategy (TA.090)
   a) Security and control requirements
   b) Access control to the modules, data, and system

c) Data integrity and audit trails
d) Passwords for terminals, operating system access, and security on networks

12) User Interface Style definition (TA.100)
    e) Design conventions
    f) Screen navigation
    g) Standard report layouts
13) Physical Database design (DB.040)
14) Application Data Conversion specification (CV.010)
    a) Data that needs to be converted
    b) Functional requirements for extracting and loading the data
    c) Requirements for validating and cleaning up both extract and load
    d) Requirements for handling the error conditions
    e) Scope
    f) Objectives
    g) Critical success factors and risks
15) Initial User Reference document (DO.060)
    a) Description of detailed functionality of each module
16) Initial User Guide (DO.070)
    a) Description of how the application system inter-relates with manual procedures
17) Initial System Operations Guide (DO.090)
    a) Explanation to System Administrators how to respond to each system operations event
18) Online Help Text (DO.140)
    a) Form-level full-screen help text
    b) Block-level full-screen help text
    c) Field-level full-screen help text
    d) Field-level hint text
    e) Error messages
    f) Error help text
19) Glossary of terms (DO.010)
20) Approval to turnover software components for test
21) Release Package (detail) sign-off
    a) Production Database DDL (DB.050)
    b) Application Code (MD.090)
    c) Code Conversion Modules (CV.050)
    d) Initial User Reference Document (DO.060)
    e) Initial User Guide Document (DO.070)
    f) Initial High-Level Technical Reference (DO.080)
    g) Initial System Operations Guide (DO.090)
    h) Online Help Text (DO.140)
22) SIT schedule from the Software Development Plan (SDP)
23) Performance assessment criteria

Outputs:

1) Application Software Test Plan (STP)
2) Documented test cases
3) Documented test scenarios / scripts
4) Test data
5) Interface test output files
6) Test environment / databases
7) SIT Test discrepancy reports (TDRs)
8) Application Test Analysis Report(s) for each test cycle
9) Application Software Test Report (STR)
10) Completed Appendix D of DCII Specification for application
11) Test metrics
12) Release Package sign-off
    a) Production Database DDL (DB.050)

- b) Application Code (MD.090)
- c) Code Conversion Modules (CV.050)
- d) Script procedures
- e) Installation instructions
- f) User Reference Document (DO.060)
- g) User Guide Document (DO.070)
- h) High-Level Technical Reference (DO.080)
- i) System Operations Guide (DO.090)
- j) Online Help Text (DO.140)
- k) Glossary of Terms (DO.010)

Task Ordering:

1) Strategy
2) Analysis
3) Design
4) Construct
5) Execute

Project Management Tasks:

1) Verify SIT testing time identified in SDP
2) Ensure SIT testing time is sufficient

Component Tasks:

Configuration Management Tasks:

1) Ensure SIT testing environment is established
2) Verify Configuration Items are correct for SIT

Component Metrics:

1) Data collected to determine size of the release to be tested:
   - a) Number of testable requirements identified by category, using CMIS and requirements traceability matrix:
     1. Requirements pertaining to entity relationships
     2. Requirements relating to the data, use of data, data values, and data rules
     3. Compliance requirements (FFMR - Blue Book certification requirements)
     4. Security requirements (System Security Plan / MSRL requirements)
     5. Application specific performance requirements
     6. Technical / system requirements
     7. System interface requirements
     8. User interface requirements
     9. Data conversion requirements
     10. Performance assessments
     11. Documentation requirements
   - b) Number of test scenarios / scripts created for each requirement
   - c) Criticality of test cases
   - d) Test data created for each test scenario / script
   - e) Number of test cycles

2) Data collected to determine the effort required to test the release using LRS:
   - a) Estimated / actual hours to perform test strategy
   - b) Estimated / actual hours to perform test analysis
   - c) Estimated / actual hours to perform test design
   - d) Estimated / actual hours to perform test construct

e) Estimated / actual hours to perform test execute

3) Data collected to determine the schedule for the release using MSProject:
   a) Estimated / actual hours to create or update scripts
   b) Estimated / actual hours to build or obtain test data
   c) Estimated / actual hours required to receive release
      1. Planned / actual start date (difference between planned and actual start date)
      2. Planned / actual completion date (difference between planned and actual completion date)

4) Data collected to specify actual schedule for the release using the Software Test Report (STR):
   a) Estimated / actual number of test scenarios / scripts and data to be created or updated
   b) Estimated / actual manpower and resources required to perform test
   c) Planned / actual tasks performed (number of test cases executed, number not executed)
   d) Tasks performed that were identified in the plan
   e) Tasks performed that were not identified in the plan
   f) Number of changes made to the test environment, and hours required to make each change
   g) Hours required to troubleshoot test environment
   h) Hours spent defect tracking or troubleshooting the application
   i) Time expended for meetings, trips, teleconferences, etc.

5) Data collected to determine quality of the release:
   a) Using CMIS, the number of valid TDRs identified in the next level of testing for release (Functional Validation Test)
   b) Using Remedy, the number of valid trouble calls that result in an identified problem report that could have been found during testing (non-operator errors)

Controls:

1) DCII Standards (web site)
   a) Defense Corporate Information Infrastructure (DCII) specification
2) DII COE Standards (web site)
   a) DII Strategic Enterprise Architectures
   b) DII COE Interface and Run-Time Specification (I&RTS)
   c) DII User Interface Specification (UIS)
3) DFAS Information Technology Architecture
   a) Technical Architecture Framework for Information Management (TAFIM)
   b) JTA Standards
4) DoD Data Administration standards
   a) Defense Data Dictionary System (DDDS)
   b) Defense Finance and Accounting Data Model (DFADM)
5) Global Combat Support System (GCSS) (Interoperability)
6) IEEE/EIA J-STD-016
7) IEEE/EIA 12207

Task Name: Develop SIT Test Strategy	**Task Number: T-SIT-001**
Component: Software Integration Test	**Category:** Software Engineering

1. **Task Name:** Develop Software Integration Test Strategy

2. **Purpose:**

   The purpose of this task is to; identify the methods and techniques to be used to validate the user defined functional requirements, define the level of software testing to be performed, identify the roles and responsibilities of the participants in the test and document the plan of action in a Software Test Plan.

   During the strategy task, the plans, schedules and resources outlined in the TEMP are reviewed and modified, as necessary. Items specific to SIT are documented in a Software Test Plan. The test plan is distributed to management for review and approval.

   Information gathered during the strategy task is documented in Sections 1.3, 1.4, 2.0, 3.0 and 4.1 of the Software Test Plan.

3. **Roles:**

   Developing the test strategy is the responsibility of the application test manager for the Software Integration Test (SIT).

4. **Entrance Criteria:**

   a. Application Business Process Model (RD.010)
   b. Application Business Data Model (RD.040)
   c. DCII specification
   d. Application Test and Evaluation Master Plan (TEMP)
   e. Application Interface specification (TA.010)
   f. Detailed System Operational requirements (TA.050)
   g. Security and control strategy (TA.090)
   h. User Interface Style definition (TA.100)
   i. Physical Database design (DB.040)
   j. Application Data Conversion specification (CV.010)
   k. Glossary of terms (DO.010)
   l. Approval to turnover software components for test
   m. SIT schedule from the Software Development Plan
   n. Performance assessment criteria

5. **Procedures:**

   The application TEMP provides the roadmap for the application's test and evaluation plans, schedules, and resource requirements. During the strategy task, the plans, schedules, and resources outlined in the TEMP are reviewed and modified, as necessary and documented within the Software Test Plan (STP). Document results of the following procedures in the sections of the Software Test Plan outlined within parentheses.

   a. Review the application Test and Evaluation Master Plan to identify the following information (Section 1.4):
      1) What is the overall objective of the application's test and evaluation
      2) Which standards are to be used in the testing process
      3) When is testing complete
      4) Which testing tools are available for the test
      5) What are the test manpower resources

6) What are the management roles and responsibilities for the test

b. Review the Software Development Plan for predefined project / test schedules (Section 1.4)

c. Identify the requirement categories to be tested (Section 1.3)
  1) Functional business requirements
  2) Requirements pertaining to entity relationships
  3) Requirements relating to the data, use of data, data values, and data rules
  4) Compliance requirements (FFMR - Blue Book certification requirements)
  5) Security requirements (System Security Plan / MSRL requirements)
  6) Application specific performance requirements
  7) Technical / system requirements
  8) Interface requirements
  9) User interface requirements
  10) Data conversion requirements
  11) Performance assessments
  12) Documentation requirements

d. Review DCII standards to determine:
  1) Are there DCII standards to be verified during software integration testing

e. Review Project standards to determine:
  1) Are there Project standards to be verified during software integration testing

f. Review the DCII Specification for compliance checklist (Appendix D)
  1) Identify the DCII requirements to be verified during SIT
  2) Identify the DCD requirements to be verified during SIT (if applicable)
  3) Identify the DCW requirements to be verified during SIT (if applicable)
  4) Identify the APP (application) requirements to be verified during SIT

g. Review the application Business Process Model to
  1) Identify the business processes pertaining to the application only
  2) Identify the business processes that interface with the application under test

h. Review the application Business Data Model to
  1) Identify the data required as input to each process
  2) Identify the data relationships between processes

i. Determine the testability of the requirements by evaluating the following:
  1) Level of detail
    A. Is the requirement specific
    B. Is the requirement measurable
  2) Methods of which the requirement may be tested
    A. Validation by use of the application only
    B. Validation using the application under test and a database query tool
    C. Validation by checking the output only
  3) Ambiguities
  4) Number of functional logic paths
  5) Number of different data types, options, and values
  6) Interfaces (inputs / outputs)
  7) Edits
  8) Number of relating database tables
  9) Number of business functions

j. Define the depth of test coverage based upon the requirements
  1) Review criticality assigned (detail) to each requirement
  2) Identify level of test coverage for each criticality (Section 4.1.1, 4.1.2, 4.1.3)

        A. Criteria for failing test
        B. Criteria for re-test beginning at the unit / module level
        C. Criteria for re-test beginning at the SIT level
        D. Criteria for successful test
    3) Choose the test techniques used to gain the required test coverage for SIT

k. Determine the necessary number of test cycles (detail) (how many iterations)
    1) Identify the percentage of the requirements with a criticality label of "high"
    2) Identify the overall critical business functions of the application

l. Confirm test resources identified in the TEMP (Section 3.x.7)
    1) Names of test resources
    2) Skill level of testing resources (detail)
    3) Dates available
    4) Identify type of training required to perform test
    5) Identify optimum time to provide training

m. Identify roles and responsibilities of all participating personnel (Section 3.x.6, 3.x.7). For each of the following, identify the actual person or persons that will perform the task.
    1) Test setup and execution:
        A. Defines test cases / scenarios
        B. Participates in the test execution
        C. Executes the test cases
        D. Supports the test execution
        E. Documents the results of the test cases
        F. Determines the test is complete
    2) Problem reporting / tracking:
        A. Tracks Test Discrepancy Reports (TDRs)
        B. Approves problems to be fixed for a release
        C. Responsible for making the software fixes
        D. Provides feedback to the TDRs
        E. Tracks which fixes within a release
    3) Test data:
        A. Provides test data for the test cases
        B. Pre-loads data for testing
        C. Controls test data
        D. Validates test data prior to test
    4) Test environment:
        A. Provides hardware and software for the test environment
        B. Establishes test environment
        C. Controls test environment
        D. Provides server support for the test environment
        E. Provides Web server support for the test environment
        F. Provides client support for the test environment
        G. Provides LAN support for the test environment
        H. Provides DBA support
            - Application level DBA support
            - Server DBA support
    5) Configuration management
        A. Provides control over the software configuration items
        B. Provides the software releases to testing, and where it is placed
        C. Provides control over test configuration items
        D. Documents the software configuration items that is impacted in order to fix a problem
    6) Management responsibilities
        A. Functional point of contact
        B. Design point of contact
        C. Responsible for determination of what test environment is needed

   D. Provides application training for test group
   E. Provides tool training for test resources
   F. Responsible for developing the SIT Software Test Report
   G. Approves for turnover of software components

n. Notify each participant of their assigned role

o. Establish Points of Contact (POC) for external organizations

p. Identify test environment (Section 3)
   1) Identify all software items needed for the test (section 3.x.1)
      A. OS
      B. Database
      C. Communication software
      D. Compilers
      E. Supporting software
   2) Identify all hardware items needed for the test (Section 3.x.2)
      A. Client hardware / server hardware / web hardware
      B. List type of equipment and version
   3) Identify printed material or processing information that must be provided to participants of the test (Section 3.x.3)
   4) Identify licenses or special permission required for software used in the test (Section 3.x.4)
   5) Identify who will provide the test environment (Section 3.x.5)
   6) Identify when test environment will be created and tested (Section 3.x.5)
   7) Identify controls for maintaining test environment (Section 3.x.5)
   8) Identify the test sites (Section 3.x)
      A. How many test environments
      B. Is the test environment different from development environment
      C. Where will the test environment(s) reside
      D. How long is the environment(s) needed
      E. What size are the environment(s)
      F. Where do I get the data for the environment(s)
      G. What security access is needed to get to the data or to perform the test
      H. What tools are needed on the test environment to perform my test
      I. Do I need special permission to use tool in the test environment

q. Identify security access for the application under test
   1) Test user accounts
   2) System administrator accounts
   3) Access privileges
   4) Role assignments
   5) Firewall security protection

r. Identify test product (detail) storage
   1) Identify each of the following:
      A. Location of storage
      B. Size required
      C. Required retention timeframe / archive requirements
      D. Access required for archived products
      E. Standards for storage (naming conventions, locations, versions)
      F. Standards for use and reuse

s. Identify procedures for resolving problems / discrepancies (Section 4.1.5)
   1) Identify information required to use CMIS for problem reporting
      A. Location / address of CMIS database
      B. Release in CMIS

C. Security access – type of CMIS user for each tester and participant in the test (i.e. Technical Action Manager, Employee, etc.)
D. Criteria used to assign a priority / criticality to a TDR (detail)
E. Testing structure to use within CMIS (system test events, system test levels)

2) Identify sequence of events required for a TDR fixed
   A. Sequence if everyone agrees TDR should be fixed
   B. Sequence if developer does not agree with tester that TDR should be fixed
   C. Sequence if TDR is to be converted to SCR
   D. Sequence if TDR is to be cancelled

t. Identify reporting criteria
   1) Identify the report standards to be used
   2) What is the frequency required for generating the report
   3) Routing of the report
   4) Who has responsibility for action items within the report

u. Define test results procedures (Section 4.1.5)
   1) How results are documented
   2) Who is notified when test is complete
   3) Who insures all software integration tests have been validated

v. Identify any externally mandated test standards
   1) Test Plan format standard
   2) Test Script format standard
   3) Test reporting format standard
   4) Required reviews to be performed

w. Build structure to capture effort metrics
   1) Identify test tasks in LRS for each application under test
      A. Strategy
      B. Analysis
      C. Design
      D. Construct
      E. Execute

6. Verification:

a. Completion of required SIT reviews

7. Exit Criteria:

a. Initial Software Test Plan (format) (Sections **1.3, 1.4, 2.0, 3.0 , 4.1.1, 4.1.2, 4.1.3, 4.1.5)**

6) Measures:

a. Data collected to determine size of the release to be tested:
   1) Number of testable requirements identified by category, using CMIS and requirements traceability matrix:
      A. Requirements pertaining to entity relationships
      B. Requirements relating to the data, use of data, data values, and data rules
      C. Compliance requirements (FFMR - Blue Book certification requirements)
      D. Security requirements (System Security Plan / MSRL requirements)
      E. Application specific performance requirements
      F. Technical / system requirements
      G. Interface requirements
      H. User interface requirements
      I. Data conversion requirements

J. Performance assessments
K. Documentation requirements
2) Criticality of test cases
3) Number of test cycles

b. Data collected to determine the effort required to test the release using LRS:
1) Estimated / actual hours to perform test strategy

b.

Task Name: Analyze SIT Requirements **Task Number: T-SIT-002**
Component: Software Integration Test **Category:** Software Engineering

1. **Task Name:** Analyze Software Integration Test Requirements

2. **Purpose:**

   The purpose of this task is to specify test cases for each feature of the application to be tested. The specified test cases provide the information required to build a test script and supporting transactions that insure the coded feature accurately implements the business requirements of the application. A result of the analysis task is an estimate of the time required to perform the test, identify the areas of the application that cannot be tested. The results of this task are documented in sections 4.2, 4.2.x, 5.0, and 6.0 of the Software Test Plan.

3. **Roles:**

   Performing the test analysis is the responsibility of the application test manager for the Software Integration Test.

4. **Entrance Criteria:**

   a. Application Business Function Model (RD.030)
   b. Test artifacts from unit/module testing
   c. Module Functional documentation (MD.040)
   d. Module Technical documentation
   e. Initial User Reference document (DO.060)
   f. Initial User Guide (DO.070)
   g. Initial System Operations Guide (DO.090)
   h. Online Help Text (DO.140)

5. **Procedures:**

   The following procedures are performed for each requirement category and specification to be validated identified in the strategy task.

   a. Decompose each type of requirement (e.g. functional, system, interface, DCII specification, etc.) into test requirements (Section 4.2)
      1) For each permutation of the requirement, define a single test requirement
         A. Analyze the requirement
         B. Define how many variations of test conditions it will take to validate the requirement
         C. Define how many variations of test conditions it will take to validate the negative conditions (edits, error handling, invalid values)
         D. Decompose each requirement into test statements or requirements
      2) Ensure there is either a one-to-one relationship or a one-to-many relationship (e.g. one business requirement to one test requirement or multiple test requirements, not multiple business requirements to one test requirement)

   b. Build traceability matrix (detail) (Section 6.0) linking each type of requirement to the test requirements
      1) Link each requirement to their associated test requirement(s)
      2) Link compliance requirements to the appropriate functional / business requirement
      3) Link DCII specification requirements to the test requirement(s)

c. Identify test data needed to support test requirements
   1) Analyze the test requirements to be validated
   2) Determine data needs for each test requirement
   3) Review available data
   4) Determine type of data that must be created versus what is available
   5) Document data needs and how to obtain data

d. Determine complexity values of test requirements
   1) Analyze the complexity value of each test requirement (detail)
   2) Analyze the test method value to be used for validating the test requirement (detail)
   3) Analyze the test data value in support of the test requirement (detail)

e. Define the test case for each test requirement (Section 4.2.x)
   1) Analyze the test requirement
   2) Define the sequence of events in order to validate the requirement
   3) Define the technical operations in support of the test (i.e. database table loaded prior to test, loading of transaction for test, etc.)
   4) Identify compliance criteria to be validated and method used to validate
   5) Identify the application user roles and access levels needed in support of the test case
   6) Define test cases that will not or can not be validated
   7) Identify user and operational documentation to be validated

f. Document test cases within the repository
   1) Define the execution sequence for the test cases
   2) Identify any dependencies that may need to be resolved in order to execute test cases
   3) Document dependencies
   4) Identify naming conventions (detail) to be used for test script development
   5) Concurrent processing of test cases

g. Estimate size
   1) Define the complexity level (detail) of each test case

h. Estimate effort (detail)
   1) Analyze skill level of each test resource
   2) Define estimated time to develop test scripts / test data from matrix
   3) Define estimated time to execute test cases from matrix
   4) Define total estimated time to execute test cases based upon number of test cycles

i. Determine the work breakdown structure / tasking by resource
   1) Identify the test cases to be assigned to each test resource
   2) Identify the test data to be built by each test resource to support the test cases
   3) Identify any additional responsibilities that will be assigned to each test resource
   4) Document tasks within a task form (format) and forward to each test resource
   5) Confirm tasking with each test resource

j. Estimate test duration for the component using the following factors (Section 5.0):
   1) Identify size
   2) Identify effort
   3) Estimate additional time in schedule for:
      A. Time delays waiting for discrepancies to be fixed
         - Compare TDR initiate date to resolution date and/or next release date
      B. Delays caused by changes to the functionality
      C. Problems in the test environment, managing the test environment and data bases, and troubleshooting automated test tools
         - Determine amount of time required to set up test environment
         - Determine amount of time required to re-set test environment between test cycles

- Determine the amount of time required to perform tasks that were not in the plan that relate to changes to the test environment
- Determine the amount of time required to perform tasks that were not in the plan that relate to test environment troubleshooting

D. Re-testing after fixes have been made
- Determine number of TDRs fixed vs. canceled or deferred
- Determine number TDRs recorded as "not fixed"

E. Time required for test support activities like version control
- Determine the amount of time required to receive application from release management

F. Delays caused by troubleshooting application
- Determine the amount of time required to perform tasks that were not in the plan that relate to defect tracking or troubleshooting application

G. Effort to report and explain the testing status
- Determine actual time required to report status

H. Documentation of test results
- Determine actual time required to report test results

I. Time required to travel, attend meetings and coordinate activities
- Determine actual time required to attend meetings or coordinate software testing activities

J. Time required to collect and record metrics for quality improvement
K. Determine number of tasks identified in the plan that were not performed

4) Execute request for test dependencies
A. Build a list identifying the test dependencies and the method for tracking in support of testing
B. Document dependency dates
C. Provide to test management

k. Build test schedule in MS Project and provide to Test Director and Project Management

l. Evaluate how software will be released to testing. Coordinate with configuration management to determine the method that the software will be released. If the software components are to be released in increments, define the test cases and requirements that will be validated in each increment.

m. Determine the risks and contingencies of performing the level of testing identified in the plan

n. Complete Software Test Plan (STP) (format). Format, validate information, document in the appropriate sections of the Software Test Plan and forward to Test Director and Project Management for review and approval. Adjust plan as necessary.

6. Verification:

a. Review and approval of final Software Test Plan

7. Exit Criteria:

a. Documented test cases
b. Approved SIT Software Test Plan

8. Measures:

a. Data collected to determine size of the release to be tested:
1) Number of test scenarios / scripts created for each requirement
2) Test data created for each test scenario / script

b. Data collected to determine the effort required to test the release using LRS:
1) Estimated / actual hours to perform test analysis

c. Data collected to determine the schedule for the release using MSProject:
   1) Estimated hours to create or update scripts
   2) Estimated hours to build or obtain test data
   3) Estimated hours required to receive release
      A. Planned start date
      B. Planned completion date

Task Name: Design Software Integration Tests	**Task Number: T-SIT-003**
Component: Software Integration Test	**Category:** Software Engineering

1. **Task Name:** Design Software Integration Tests

2. **Purpose:**

   The purpose of software integration test design is to identify the different types of transactions to be created for each test case, identify/define test data used to validate test cases, identify pre-conditions for the test environment to execute the test and define the sequence of events that must occur before and after testing.

3. **Roles:**

   The person designated to perform the software integration test design is each test resource assigned to the application under test. Overall coordination responsibility rests with the application test manager.

4. **Entrance Criteria:**

   a. Documented test cases
   b. Access to test environment
   c. Software Test Plan
   d. Release Package sign-off
   e. Application Business Process Model (RD.010)
   f. Application Business Function Model (RD.030)
   g. Application Business Data Model (RD.040)
   h. Module Functional documentation (MD.040)
   i. Module Technical documentation

5. **Procedures:**

   a. Insure that all unit / module testing has been completed

   b. Identify the different types of test transactions required for each test case
      1) Review the test cases documented for the Software Integration Test
      2) Review the business data model, business function model, business process model, module functional documentation, and module technical documentation to determine:
         A. What data is mandatory (not null), and what data is optional (null)
         B. What are the allowable values
         C. What are the methods of data selection or entry
         D. What is the behavior of the data selections (pick lists, menus, group boxes)
         E. Are there different types of data (disbursements, collections, accounting reports, etc.) to be processed
         F. What are the sources of obtaining the test data (users, subject matter experts, table data from systems, etc.)
         G. What are the different user types that have access to the data
         H. What type of access does each user type have
         I. Events that occur and their characteristics
         J. Processes that respond to or are triggered by each event

   c. Determine the sequence of execution
      1) Determine the sequence of test cases to be validated. Considerations for ordering the test cases:
         A. Normal processing sequence
         B. Sequence that will require the least amount of test data creation and / or manipulation

C. Sequence to make it easier to validate test results

d. Track (by hour) effort for each test case in MsProject for:
   1) Building of test scenarios / scripts
   2) Defining test data in support of test cases

e. Build a matrix that identifies the following items
   1) The logical order of execution of the test cases
   2) Roles/accesses of users that sent and receive data for the test case
   3) Data that is sent to and received from the test case
   4) Dependencies between one test case and another
   5) Concurrent processing of test cases

f. Define data for test database and non-database files
   1) Review the test data available to perform the test case
   2) Build or locate the test data required to validate the test case
      A. Valid transaction data
      B. Invalid transaction data
      C. Interface data
      D. Supporting data
      E. Static table data
   3) Locate or build input test data files, if necessary
   4) Obtain table data from interfacing software/systems, if applicable
   5) Build spreadsheets/tables or entity instance charts (detail) with test data to be entered through the user interface during test

g. Design a method that user documentation will be validated such as online Help text, user manuals, and software training material. Document within test cases.

h. Conduct a peer review of the test design

i. Conduct a test design review with functional and technical experts for approval
   1) Adjust test design as necessary

6. Verification:

a. Review and approval of test design

7. Exit Criteria:

a. Test design (execution order, test cases / scenarios)
b. Test data required to validate test cases

8. Measures:

a. Data collected to determine the effort required to test the release using LRS:
   1) Estimated / actual hours to perform test design

Task Name: Construct Software Integration Tests	**Task Number: T-SIT-004**
Component: Software Integration Test	**Category:** Software Engineering

1. **Task Name:** Construct Software Integration Tests

2. **Purpose:**

   The purpose of Software Integration Test construct task is to build the test scripts used during the execution of the test. Test scripts are based upon the test cases.

3. **Roles:**

   Constructing the test scripts is the responsibility of each test resource assigned to the application under test.

4. **Entrance Criteria:**

   a. Documented test cases
   b. Test data
   c. Access to test environment

5. **Procedures:**

   Each procedure during the construction task requires actual effort by test resources to perform. Document effort (by hours) in MSProject for each test case. If building test scripts and/or test data for a test case, document the time expended towards the test case that relates to the scripts / data. Document time for peer reviews, meetings, and building the test environment in MSProject relating to that specific task.

   a. Build manual test scripts (format). The script should include:
      1) Test case to be validated
      2) Data to be used in the validation
      3) Pre-setting or staging condition that must happen prior to the validation
      4) Steps required to run the test
      5) Method used to review results and verify what is expected
      6) Expected results
      7) Reference to test requirement being validated

   b. Build manual test scenarios (master test script)
      1) Identify sequence of scripts
      2) Identify testing that can be executed concurrently

   c. Automate scripts (where possible)

   d. Automate scenarios to execute the same sequence as manual scripts

   e. Update requirement traceability matrix (document the following crosswalks)
      1) Test scripts to test requirements
      2) Test scripts to compliance requirements

   f. Build test environment
      1) Setup hardware
      2) Build database instance (DBA function)
      3) Populate data base tables with baseline data
      4) Build / acquire interface data files

5) Establish security access
6) Install application under test
7) Build software support (stubs/ drivers)
8) Load input test data interfaces and validate load
9) Load actual input interfaces to be used for test
10) Backup / export baseline test data

g. Perform final test peer review
1) Verify environment, test scripts, and test data are ready for commencement of test
2) Review order of execution of test cases and simultaneous testing
3) Adjust for test as necessary within the project timeline

6. Verification:

a. Peer review of work products

7. Exit Criteria:

a. Manual test scripts / scenarios
b. Automated test scripts / scenarios
c. Input interface test files
d. Test environment
e. Supporting software
f. Updated requirements traceability matrix
g. Populated baseline test database

8. Measures:

a. Data collected to determine the effort required to test the release using LRS:
1) Estimated / actual hours to perform test construct
b. Data collected to determine the schedule for the release using MSProject:
1) Actual hours to create or update scripts
2) Actual hours to build or obtain test data

Task Name: Execute Software Integration Tests	**Task Number: T-SIT-005**
Component: Software Integration Test	**Category:** Software Engineering

1. **Task Name:** Execute Software Integration Tests

2. **Purpose:**

   The purpose of this task is to execute test scripts that validate the test cases defined in the analysis task, ordered in the design task, and built in the construct task. During the test execution task the test resource will compare the expected results to the actual test results, formally report any noted discrepancies, validate that the software meet the requirement, and identify any potential risk within the software.

3. **Roles:**

   Each test resource assigned to the application under test is responsible for test execution, and regression test execution. The test manager is overall responsible for the execution of each test cycle.

4. **Entrance Criteria:**

   a. Manual test scripts / scenarios
   b. Automated test scripts / scenarios
   c. Access to test database
   d. Access to test environment
   e. Access to CMIS

5. **Procedures:**

   Each procedure during the execution task requires actual effort by test resources to perform. Document effort (by hours) in MSProject for each test case. If reworking test scripts and/or test data for a test case, document the time expended towards the test case that relates to the scripts / data. Document time for the test execution in MSProject for each test case, by test cycle.

a. Document start date in CMIS and the test scenario (master script)

b. Turn on operational monitoring tools if coordinated with technical personnel

c. Execute test scripts
   1) Record the start date / time for the test script
   2) Execute the steps within the test script
   3) Compare the actual results with the expected results and note any discrepancies
   4) During execution, capture proof of validation for compliance criteria such as GUI screen captures, database table updates, audit records
   5) Record completion date / time for the test script
   6) Document the results of the test script
   7) Modify / correct test script, if necessary
   8) Run all steps listed on master test script in order specified

d. Generate Test Discrepancy Reports (TDRs) (detail) for any noted discrepancies from the test results
   1) Record TDRs in CMIS according to TDR procedures (web site)
   2) Document the sequence of events within the TDR
   3) Document the difference between the expected results and the actual test results
   4) Document the suspected problem
   5) Document any test data that was used within the test

6) Document the test script identification number that the discrepancy was found
7) Assign a priority level to the TDR
8) Save and route TDR according to discrepancy reporting procedures

e. Record TDRs within the TDR log (format)

f. Document test complete date in CMIS of the test scenario (master test script)

g. Build a “Quick Look” test summary report (format)
   1) Document work-arounds, discrepancies found, fixes, and risks
   2) Route to the Test Director and Project Management
   3) Track any open items for Project Management for decision of fixes for next release
   4) Record the amount of effort completed to date, and remaining amount of effort

h. Re-test as necessary
   1) Reset test environment
   2) Coordinate with the Release Management for a new release
   3) Install new release
   4) Repeat test cycle as specified in the test plan
   5) Validate TDRs fixes
   6) Track TDRs to resolution
      A. All TDRs should be resolved at the end of the test event (cancel, defer or fix)

i. Document Software Test Report (STR) (format)
   1) Document work-arounds, discrepancies found, fixes, and risks
   2) Include evaluation of software application quality within report
   3) Route to the Test Director and Project Management
   4) Track that any remaining TDRs are converted to SCRs or cancelled
   5) Document total effort and quality measurements

j. Complete Appendix D of the DCII Specification for applicable requirements

k. Clean up test products (detail)
   1) Correct / update test scripts as needed
   2) Correct / update test data as needed
   3) Automate regression test scripts / scenarios (this is a long term effort that does not have to be completed before the SIT is considered complete)
   4) Review metrics to insure they are complete and accurate

l. Place test products under Configuration Management control (this is a long term effort that does not have to be completed before the SIT is considered complete)

m. Provide process improvements to process owner.

6. Verification:

a. Updated test script documentation with test completion date

7. Exit Criteria:

a. Software Test Report
b. Updated test scripts
c. Automated regression test
d. Release Package
e. DCII Specification Appendix D (appropriate portions completed)
f. Test Discrepancy Reports (TDRs)
g. Process improvement suggestions

8. Measures:

a. Data collected to determine the effort required to test the release using LRS:
   1) Estimated / actual hours to perform test execute

b. Data collected to determine the schedule for the release using MSProject:
   1) Actual hours required to receive release
      A. Planned / actual start date (difference between planned and actual start date)
      B. Planned / actual completion date (difference between planned and actual completion date)

c. Data collected to specify actual schedule for the release using the Software Test Report (STR):
   1) Estimated / actual number of test scenarios / scripts and data to be created or updated
   2) Estimated / actual manpower and resources required to perform test
   3) Planned / actual tasks performed (number of test cases executed, number not executed)
   4) Tasks performed that were identified in the plan
   5) Tasks performed that were not identified in the plan
   6) Number of changes made to the test environment, and hours required to make each change
   7) Hours required to troubleshoot test environment
   8) Hours spent defect tracking or troubleshooting the application
   9) Time expended for meetings, trips, teleconferences, etc.

d. Data collected to determine quality of the release:
   1) Using CMIS, the number of valid TDRs identified in the next level of testing for release (Functional Validation Test)